

# An Evaluation of Coupling Metrics for Aspect-Oriented Software

Haihao Shen and Jianjun Zhao

School of Software  
Shanghai Jiao Tong University  
800 Dongchuan Road, Shanghai 200240, China  
{haihaoshen,zhao-jj}@sjtu.edu.cn

**Abstract.** Coupling is an internal software attribute that can be used to indicate the degree of system interdependence among the components of a software. Coupling is thought to be a desirable goal in software construction, leading to better values for maintainability, reusability and reliability. Although several coupling frameworks and coupling metrics have been proposed for aspect-oriented software, the tool support and empirical evaluation of these metrics are still being missed. In this paper we present our evaluation for a coupling metrics suite for aspect-oriented software. To do so, we first select a set of coupling metrics which consists of 16 coupling metrics for aspect-oriented software. We then develop a coupling metric tool to calculate these metrics for evaluation. Finally we evaluate these metrics by means of correlation analysis between coupling metrics and external attributes.

## 1 Introduction

Aspect-oriented programming (AOP) [11] is a brand new methodology of programming. It overcomes the weakness of object-oriented programming (OOP) when modeling the crosscutting concerns. By extracting crosscutting concerns and encapsulating them into separate programming module, AOP eliminates code tangling in object-oriented programs and improves the modularity of OO programs, making aspect-oriented software more maintainable, extensible and flexible.

While AOP makes the crosscutting concerns separate, it also brings the new features which make it difficult to measure the complexity of AO software. Until now, although several coupling frameworks and coupling metrics have been proposed for aspect-oriented software, the tool support and empirical evaluation of these metrics are still being missed. If a coupling framework is lack of the tool support, the coupling metrics defined in the coupling framework are not practically but theoretically meaningful. As a result, people can not understand what effect these coupling metrics have on AO software actually. With respect to the coupling framework of AO software, there is no specified metric tool for the specified coupling framework to compute the metrics. Although Ceccato [8] proposes the coupling framework and the common metrics, no strong empirical

evaluation could be finished due to no metric tool. Further research into AO coupling frameworks has been carried by Bartsch and Harrison [5] and an empirical validation of AO coupling measures has been proposed. However, the lack of metric tool support is also a problem for the empirical validation. Therefore, there is a strong need for tool support and empirical evaluation of aspect-oriented coupling metrics.

In this paper we present our evaluation for a coupling metrics suite for aspect-oriented software. We set the type of connection as our criterion for selecting candidate coupling metrics for the further evaluation. Our suite consists of nine common coupling metrics selected from existing coupling metric frameworks and seven new coupling metrics defined by us in this paper. Based on the metrics suite, we develop the tool called **AJMetrics** to measure these metrics quantitatively. Moreover, we analyze the correlation between coupling metrics and external attributes to evaluate the coupling metrics and make several conclusions about what coupling metric has the most effect on the AO software and what external attribute is more important to the coupling metrics. At last, we use an empirical sample written in AspectJ [3] to verify the correctness of the evaluation result. To the best of our knowledge, our work is the first attempt to automatically evaluate what coupling metrics have the most effect on AO software and what external attributes are more important to coupling metrics by analyzing the measurement results based on several tools.

The rest of this paper is organized as follows. Section 2 introduces our coupling metrics suite for AO software. Section 3 presents our coupling metric tool **AJMetrics**. Section 4 presents our evaluation of coupling metrics and verifies the evaluation result. Section 5 discusses related work. Concluding remarks are given in Section 6.

## 2 A Coupling Metrics Suite for AO Software

We next present our coupling metrics suite used for evaluation. Briand *et al.* [7] propose a coupling framework for object-oriented systems which consists of six criteria. Each criterion determines one basic aspect of the resulting measure. The six criteria of the framework are: *type of connection*, *locus of impact*, *granularity of the measure*, *stability of server*, *direct or indirect coupling*, and *inheritance*. In this paper, due to space limitation, we only take the criterion of the type of connection as our coupling metric selection criterion for our evaluation. With respect to the coupling metrics suggested by Ceccato [8], our metrics suite includes some common coupling metrics which have been included in several existing coupling metric frameworks in our coupling metrics suite. Our suite also contains seven new coupling metrics defined by us in this paper.

### 2.1 Common Coupling Metrics

According to the existing criterion and coupling metrics for AO systems, we make a further study into the coupling frameworks and coupling metrics. Among these

coupling frameworks, several coupling metrics from different coupling frameworks are similar in the logogram and definition and we call them common coupling metrics, and include them into our metrics suite.

1. **DIT (Depth of Inheritance Tree)**: DIT measures the length of the longest path from a given module to the class/aspect hierarchy root. The deeper a class/aspect is in the hierarchy, the greater the number of operations it might inherit, thus making it more complex to understand and change.
2. **NOC (Number Of Children)**: NOC measures the number of immediate subclasses or sub-aspects of a given module. The number of children of a module indicates the proportion of modules potentially dependent on properties inherited from the given one.
3. **CFA (Coupling on Field Access)**: CFA measures the number of modules or interfaces declaring fields that are accessed by a given module. This metric considers the dependences of a given module on other modules, in terms of accessed fields.
4. **CMC (Coupling on Method Call)**: CMC measures the number of modules or interfaces declaring methods that are possibly called by a given module. This metric considers the dependences of a given module on other modules, but in terms of methods, instead of accessed fields.
5. **CBM (Coupling between Modules)**: CBM measures the number of modules or interfaces declaring methods or fields that are possibly called or accessed by a given module. This metric is split into two metrics (CFA and CMC) to distinguish coupling on operations from coupling on attributes.
6. **CDA (Crosscutting Degree of an Aspect)**: CDA measures the number of modules affected by the pointcuts and by the intertype-declarations in a given aspect. All modules possibly affected by an aspect explicitly or implicitly will be taken into account. This gives an idea of the overall impact an aspect has on the other modules.
7. **CAE (Coupling on Advice Execution)**: CAE measures the number of aspects containing advices possibly triggered by the execution of operations in a given module. This metric considers the implicit dependence of the operation from the advice if the behavior of an operation can be altered by an aspect advice, due to a pointcut intercepting it.
8. **CIM (Coupling on Intercepted Modules)**: CIM measures the number of modules or interfaces explicitly named in the pointcuts belonging to a given aspect. This metric takes into account only those modules and interfaces an aspect is aware of those that are explicitly mentioned in the pointcuts.
9. **RFA (Response For a Module All)**: RFA measures the number of methods and advices potentially executed in response to a message received by a given module. This metric computes the implicit responses that are triggered whenever a pointcut intercepts an operation of the given module.

## 2.2 New Coupling Metrics

According to the type of connection, we further propose seven new coupling metrics as follows.

1. **CDP (Crosscutting Degree of an Aspect caused by Pointcuts)**: CDP measures the number of modules affected by the pointcuts in a given aspect. This metric is split from CDA. Thus, this gives an idea of the overall impact an aspect has on the other modules caused by pointcuts.
2. **CDI (Crosscutting Degree of an Aspect caused by Intertype-declarations)**: CDI measures the number of modules affected by the intertype-declarations in a given aspect. This metric is also split from CDA. Thus, this gives an idea of the overall impact an aspect has on the other modules caused by intertype-declarations.
3. **CAM (Coupling on Advice Execution caused by Methods)**: CAM measures the number of aspects containing advices possibly triggered by the execution of methods in a given module. This metric is split from CAE. This metric considers the implicit dependence of the operation from the advice if the behavior of an operation can be altered by an aspect advice caused by methods.
4. **CAA (Coupling on Advice Execution caused by Advices)**: CAA measures the number of aspects containing advices possibly triggered by the execution of advices in a given module. This metric is also split from CAE. This metric considers the implicit dependence of the operation from the advice if the behavior of an operation can be altered by an aspect advice caused by advices.
5. **CAI (Coupling on Advice Execution caused by Intertype-declarations)**: CAI measures the number of aspects containing advices possibly triggered by the execution of intertype-declarations in a given module. This metric is also split from CAE. This metric considers the implicit dependence of the operation from the advice if the behavior of an operation can be altered by an aspect advice caused by intertype-declarations.
6. **RFM (Response For a Module caused by Methods)**: RFM measures the number of methods potentially executed in response to a message received by a given module. This metric is split from RFA and computes the implicit responses that are triggered whenever a pointcut intercepts methods of the given module.
7. **RFP (Response For a Module caused by Pointcuts)**: RFP measures the number of advices potentially executed in response to a message received by a given module. This metric is also split from RFA and computes the implicit responses that are triggered whenever pointcuts intercepts an advice of the given module.

Note that choosing a type of connection implies choosing the mechanism that constitutes coupling among aspects and classes. Table 1 and Table 2 summarize the types of connection of the common and new coupling metrics. In Table 1 and Table 2 there are links between “Client Item” and “Server Item”. “Client Item” may be aspects, classes, pointcuts, intertype-declaration, advices, or methods while “Server Item” may be fields, methods, advices, aspects, or classes. Usually, “Client Item” of the “Client Class” should access “Server Item” of the “Server Class”. Moreover, “Class(I)” means it may be a class or an interface.

**Table 1.** Common Coupling Metrics

#	Metric	Client Class	Client Item	Server Item	Server Class
1	DIT	Aspect,Class	Aspects,Classes	Hierarchy Path	Aspect,Class
2	NOC	Aspect,Class	Aspects,Classes	Subclasses, Subaspects	Aspect,Class
3	CFA	Aspect,Class	Aspects,Classes	Fields	Aspect,Class(I)
4	CMC	Aspect,Class	Aspects,Classes	Methods	Aspect,Class(I)
5	CBM	Aspect,Class	Aspects,Classes	Methods,Fields	Aspect,Class(I)
6	CDA	Aspect	Pointcuts, Intertype- declarations	Aspects,Classes	Aspect
7	CAE	Aspect,Class	Methods,Advices, Intertype- declarations	Advices	Aspect,Class(I)
8	CIM	Aspect	Pointcuts	Aspects,Classes	Aspect,Class(I)
9	RFA	Aspect,Class	Pointcuts,Methods	Methods,Advices	Aspect,Class

**Table 2.** New Coupling Metrics

#	Metric	Client Class	Client Item	Server Item	Server Class
1	CDP	Aspect	Pointcuts	Aspects,Classes	Aspect,Class
2	CDI	Aspect	Intertype- declarations	Aspects,Classes	Aspect,Class
3	CAM	Aspect,Class	Methods	Advices	Aspect
4	CAA	Aspect,Class	Advices	Advices	Aspect
5	CAI	Aspect,Class	Intertype- declarations	Advices	Aspect
6	RFM	Aspect,Class	Methods	Methods,Advices	Aspect,Class
7	RFP	Aspect,Class	Pointcuts	Methods,Advices	Aspect,Class

### 3 AJMetrics: A Coupling Metric Tool for AO Software

We next present our coupling metric tool **AJMetrics** for calculating the values of metrics. **AJMetrics** is the abbreviations for the metric tool “AspectJ Metrics tool” and based on the open-source project AopMetrics [2].

#### 3.1 Implementation Issues

Several open-source tools can be used to compute the value of coupling metrics. The tool AopMetrics is a common metrics suite tool for Java and AspectJ. Another tool MuLATO [13] stands for multi-language assessment tool. It is a Java application for collecting metrics from programs written in several languages such as Java and AspectJ. Since this paper mainly discusses the coupling

framework of AO software written in AspectJ, we choose AopMetrics as a basis for `\verbAJMetrics++` implementation. `\verbAJMetrics++` can compute both common metrics and those new metrics we proposed in this paper. We implement the new metrics based on the information of the structure of AST for an AspectJ program. Next, much more implementation details are as follows:

- **Listing all the files in overall system**

Firstly, `AJMetrics` traverses all the AspectJ files and Java files in the hierarchy of packages in overall system, and then generates the “.lst” file which consists of package name and filename for each AspectJ file and Java file.

- **Compiling and Generating Abstract Syntax Tree structure**

Secondly, `AJMetrics` compiles the overall system according to the “.lst” file, and then generates AST structure for system.

- **Measuring coupling metrics for each component**

Thirdly, `AJMetrics` measures the coupling metrics for each component (AspectJ file or Java file) by means of collecting the coupling information according to the AST structure.

### 3.2 Sample Test

The sample under test is an open-source project called `Glassbox` [10], which is written in AspectJ and Java. To date, it has experienced several versions and the source code in the repository has been changed a lot. Table 3 shows the scale of `Glassbox` for three versions. Thus, we just choose an arbitrary version (06.4.17) for our test. We use our tool `AJMetrics` to compute the values of coupling metrics included in our suite and the tool generates the measurement result automatically. Fig. 1 shows a part of the coupling measurement result for the sample test.

**Table 3.** The Scale of Glassbox

Version	LOCC	Classes	Aspects	Proportion of Aspects
06.4.17	33281	207	31	13.025%
07.3.30	16840	198	28	12.389%
07.5.1	16840	198	28	12.389%

## 4 Evaluating Coupling Metrics

We next analyze the correlation between internal attributes and external attributes and evaluate the coupling metrics for AO software. After measuring the metrics, empirical evaluation of coupling metrics for AO software is needed since there are internal attributes and external attributes which are correlated and have an effect on AO software.

```

- <type kind="aspect" name="BeanConfigurator">
  <metric name="LOCC" value="15" />
  <metric name="DIT" value="0" />
  <metric name="NOC" value="0" />
  <metric name="CFA" value="0" />
  <metric name="CMC" value="1" />
  <metric name="CBM" value="1" />
  <metric name="CDA" value="0" />
  <metric name="CDP" value="0" />
  <metric name="CDI" value="0" />
  <metric name="CAE" value="0" />
  <metric name="CAM" value="0" />
  <metric name="CAA" value="0" />
  <metric name="CAI" value="0" />
  <metric name="CIM" value="0" />
  <metric name="RFA" value="2" />
  <metric name="RFM" value="2" />
  <metric name="RFP" value="0" />
</type>

```

Fig. 1. The Measurement Result For Sample Test

#### 4.1 Internal Attributes

Internal attributes mean that the attributes which can be used to indicate the degree of interdependence among the components of a software. In this paper, all of the coupling metrics are internal attributes, which include DIT, NOC, CDP, and CDI etc. Each of them shows the different interdependence among the components of a software. The values of these internal attributes can be measured by the metric tool *AJMetrics*. These values of coupling metrics are used when analyzing the correlation between internal attributes and external attributes.

#### 4.2 External Attributes

External attributes mean that the attributes which can be used to indicate the basic feature of the component itself. In this paper, there are two external attributes that can be used when analyzing the correlation between internal attributes and external attributes. They are:

- **LOCC (Lines of Class Code):**  
LOCC measures the number of perfect lines of class code excluding the blank line and comment line.
- **AGE (Days of Code being the Repository):**  
AGE measures the days of source code being the repository.

### 4.3 Correlation Analysis

In our case, we analyze the correlation between internal attributes and external attributes. Of course, correlation analysis needs the analysis tool support. Of all of the analysis tool, we choose SPSS [14] as our analysis tool. We use SPSS to analyze the Bivariate Correlation, one kind of correlation, for all the statistical values of the attributes. Since we could not obtain the entire CVS tree for the project **Glassbox**, the source code we check out from CVS is not a normal distribution. Thus, the nonparametric Spearman rank correlation is used in the Bivariate Correlation.

### 4.4 Empirical Evaluation

First, we check out the source code from **Glassbox**'s CVS, and we choose three arbitrary versions: 06.4.17, 07.3.30, and 07.5.1. The scale of **Glassbox** for three versions can be seen in Table 3. Second, we use the metric tool **AJMetrics** to calculate the values of the internal and external attributes of these three versions. The value of LOCC could be measured by **AJMetrics** as well. The value of AGE could be obtained by means of comparing the same files between different versions by the tool WinMerge [15]. Third, by the tool WinMerge, we could find those different files which have changed from one version to another version. Table 4 shows the different values of AGE between the different versions. From Table 4, we can easily infer the value of AGE between version 07.3.30 and version 07.5.1. For example, the AGE of the file "InitializeLog.java" is 70 and the AGE of the file "DetectionUtils.java" is 159.

**Table 4.** Different Files List

#	Filename	AGE	
		06.4.17 vs 07.3.30	06.4.17 vs 07.5.1
1	BootstrapLog.java	0	0
2	InitializeLog.java	136	206
3	ApplicationLifecycleAware.java	0	0
4	AutoInitialization.aj	0	0
5	BeanConfigurator.aj	0	0
6	GlassboxInitializer.aj	106	106
7	NondelegatingURLClassLoader.java	0	0
8	DetectionUtils.java	60	219

After getting the changing file list of different versions, we compute the values of coupling metrics and LOCC of the source code whose filename is in the list. Then, we analyze the correlation between internal attributes and external attributes.

Fig. 2-4 show the values of coupling metrics and LOCC in different versions respectively. Since the values of several coupling metrics are zero, these coupling metrics are not listed. The reasons are as follow:



	LOCC	DIT	NOC	CFA	CMC	CBM	RFM	RFP	RFA
1	129.0	.00	.00	1.00	4.00	5.00	26.0	.00	26.0
2	119.0	.00	.00	.00	2.00	2.00	11.0	.00	11.0
3	5.00	.00	.00	.00	.00	.00	.00	.00	.00
4	16.00	1.00	.00	.00	.00	.00	1.00	.00	1.00
5	15.00	.00	.00	.00	1.00	1.00	2.00	.00	2.00
6	166.0	.00	1.00	.00	6.00	6.00	17.0	.00	17.0
7	41.00	3.00	.00	.00	.00	.00	2.00	.00	2.00
8	16.00	.00	.00	.00	.00	.00	2.00	.00	2.00

Fig. 2. Measurement Result for 06.4.17

	LOCC	DIT	NOC	CFA	CMC	CBM	RFM	RFP	RFA
1	129.0	.00	.00	1.00	4.00	5.00	26.0	.00	26.0
2	205.0	.00	.00	1.00	2.00	3.00	15.0	.00	15.0
3	5.00	.00	.00	.00	.00	.00	.00	.00	.00
4	16.00	1.0	.00	.00	.00	.00	1.00	.00	1.00
5	15.00	.00	.00	.00	1.00	1.00	2.00	.00	2.00
6	186.0	1.0	1.00	.00	6.00	6.00	20.0	.00	20.0
7	41.00	3.0	.00	.00	.00	.00	2.00	.00	2.00
8	23.00	.00	.00	.00	.00	.00	3.00	.00	3.00

Fig. 3. Measurement Result for 07.3.30

	LOCC	DIT	NOC	CFA	CMC	CBM	RFM	RFP	RFA
1	129.0	.00	.00	1.00	4.00	5.00	26.0	.00	26.0
2	214.0	.00	.00	1.00	2.00	3.00	15.0	.00	15.0
3	5.00	.00	.00	.00	.00	.00	.00	.00	.00
4	16.00	1.0	.00	.00	.00	.00	1.00	.00	1.00
5	15.00	.00	.00	.00	1.00	1.00	2.00	.00	2.00
6	186.0	1.0	1.00	.00	6.00	6.00	20.0	.00	20.0
7	41.00	3.0	.00	.00	.00	.00	2.00	.00	2.00
8	29.00	.00	.00	.00	.00	.00	5.00	.00	5.00

Fig. 4. Measurement Result for 07.5.1

- The values of these coupling metrics, zero, will not have any effect on the correlation between internal attributes and external attributes.
- As the test example we have taken is lack of enough interdependence among the components, the values of these coupling metrics are omitted.
- As the files in the test example are not in the changing list, the coupling metrics of these files are not measured.

#### 4.5 Results

The analysis tool can help us to analyze the correlation between several coupling metrics and external attributes by means of Spearman analysis which is a typical kind of correlation analysis. First, we analyze correlation between LOCC and each coupling metric in Fig. 2-4. Table 5 describes the result of correlation analysis between different versions. Note that (1)“X” indicates that this metric has no effect on the specified Sig<sup>1</sup> and (2) “/” indicates that SPSS can not analyze

<sup>1</sup> Sig means the value of significance of the correlation coefficient.

the correlation according to the given data set and (3) if the value of P-value<sup>2</sup> of the coupling metric is smaller than the value of Sig, the metric is important.

**Table 5.** Correlation Analysis Between Different Versions

#	Metrics	Spearman coefficient			P-value			Sig		
		06.4.17	07.3.30	07.5.1	06.4.17	07.3.30	07.5.1	06.4.17	07.3.30	07.5.1
1	DIT	-0.039	0.165	0.165	0.927	0.696	0.696	X	X	X
2	NOC	0.581	0.412	0.412	0.131	0.310	0.310	X	X	X
3	CFA	0.415	0.630	0.630	0.307	0.094	0.094	X	X	X
4	CMC	0.753	0.672	0.672	0.031	0.068	0.068	0.05	X	X
5	CBM	0.753	0.672	0.672	0.031	0.068	0.068	0.05	X	X
6	RFM	0.884	0.826	0.826	0.004	0.011	0.011	0.01	0.05	0.05
7	RFP	/	/	/	/	/	/	X	X	X
8	RFA	0.884	0.826	0.826	0.004	0.011	0.011	0.01	0.05	0.05

After analyzing the correlation between the coupling metrics in Fig. 2-4 and LOCC, there are four viewpoints as follows:

- Of all the coupling metrics in Fig. 2-4, only the value of P-value for RFM and RFA is always smaller than the value of Sig.
- Of all the coupling metrics in Fig.2-4, the value of P-value for CMC and CBM is sometimes smaller than the value of Sig.
- Of all the coupling metrics in Fig. 2-4, only the value of P-value for RFP is always in the state of not being analyzed.
- With the constantly updated version of **Glassbox**, the value of P-value of NOC, CMC, CBM, RFM, RFA tends to increase while the value of P-value of DIT, CFA tends to decrease.

According to the above four viewpoints, we further make a deep conclusion about the effect these coupling metrics have on AO software. There are three conclusions as follows:

- Of all the coupling metrics, RFM and RFA have the most important effect on the AO software as in the correlation analysis of three versions, only the value of P-value of RFM and RFA is always smaller than the value of Sig. According to the definition of RFM and RFA in this paper, we could find that potential responses among components of the system are relatively smooth to the entire system. However, CBM and CMC is just the opposite. In the early version, the value of P-value is still in the important level. With the update of the version, the value is increasing which indicates that the coupling interdependence among components is increasing and the importance

<sup>2</sup> P-value means the value of the correlation level, compared to Sig.

is decreasing. Therefore, CBM is sometimes important to the AO software.

- Of all the coupling metrics, RFP has the least effect on the AO software. Maybe the data set we use to analyze the correlation is not comprehensive enough and not objective enough, and as a result, SPSS could not analyze the correlation between the coupling metrics and LOCC correctly. However, we still believe that RFP is the least important to the AO software since in the correlation analysis of three versions, the value of P-value of RFP is always zero.
- With the constantly updated version of **Glassbox**, NOC, CMC, CBM, RFM, RFA have the less effect on the AO software while DIT, CFA have the more effect on the AO software. From another point of view, developers maybe improve the code reusability and readability in the update of the versions and make the classes with the deeper depth of hierarchy and the aspects accessing the private variables in the original program decrease.

Since RFM is split from RFA and CMC is split from CBM, according to the above viewpoints, we further analyze the correlation between RFA, CBM and LOCC, AGE. Fig. 5 shows the data set including LOCC, AGE, CBM, and RFA.

	LOCC	AGE	CBM	RFA
1	86.00	136.00	1.00	4.00
2	20.00	106.00	.00	3.00
3	7.00	60.00	.00	1.00
4	95.00	206.00	1.00	4.00
5	20.00	106.00	.00	3.00
6	13.00	219.00	.00	3.00

**Fig. 5.** Further Archiving Data set

By the tool SPSS, we can analyze the correlation between RFA, CBM and LOCC, AGE respectively. The analysis result can be seen from Table 6. Note that RFA2LOCC means the correlation between RFA and LOCC. From the analysis, we can get two viewpoints:

**Table 6.** Analysis Result For Archiving Data set

#	Metrics	Spearman coefficient	P-value	Sig
1	RFA2LOCC	0.939	0.005	0.01
2	RFA2AGE	0.626	0.183	X
3	CBM2LOCC	0.840	0.036	0.05
4	CBM2AGE	0.420	0.407	X

- In the correlation analysis of RFA2LOCC and CBM2LOCC, it proves again that RFA and CBM are important to the AO software. Of course, RFA has more importance. Therefore, RFA is the most important metric among all the coupling metrics.
- In the correlation analysis of RFA2AGE and CBM2AGE, it proves that RFA and CBM are affected less by AGE.

Based on the above viewpoints, we can make the following conclusions about the effect which LOCC and AGE have on coupling metrics:

- The two coupling metrics, RFA and CBM, which are the most important to the AO software are affected less by AGE. In another words, with the update of versions, the days of the original source code being the CVS repository are increasing, but it does not has any effect on the values of the coupling metrics.
- Since AGE has less effect on RFA and CBM and LOCC has more effect on RFA and CBM, and LOCC, AGE are both the important external basic attributes, LOCC is more important to the coupling metrics than AGE.

#### 4.6 Verification

According to the correlation analysis, we have concluded some results about the evaluation of coupling metrics. Among them, RFA has the most effect on the AO software while RFP has the least. Between the external attributes, LOCC is more important to coupling metrics than AGE. However, empirical evaluation of coupling metrics for AO software needs more sample tests. Next, we use another sample test called *Ajhsqldb* [1] to verify the evaluation results. Table 7 shows the scale of *Ajhsqldb*. Version 1, version 9 and version 18 of *Ajhsqldb* are taken as examples in the verification test.

**Table 7.** The Scale of *Ajhsqldb*

Version	LOCC	Classes	Aspects	Proportion of Aspects
version 1	110102	250	7	2.723%
version 9	110325	234	11	4.489%
version 18	84151	235	25	9.615%

First, we use **AJMetrics** to calculate the values of coupling metrics of all three versions and three result files are generated respectively. Then, we use WinMerge to list the different files between the different versions. After analyzing the result files and the differences between different versions, we find that most of the results of evaluation of coupling metrics are correct.

Of all the different files between version 1 and version 9, we choose several files randomly to explain the correctness of the evaluation.

Comparing the values of RFA and RFP of all the files between version 1 and version 9, we find that the values of RFA are the same and the values of RFP are increasing with the update of the version. Since the values of RFA of some files are same, some features in the source code which affect the value of RFA are actually stable enough to the entire project. However, those files of which the value of RFP are changeable are not stable enough to the project. In a word, RFA is stable and has the most effect on the AO software while RFP is increasingly changeable and has the least effect on the AO software. Also, we find that the files in version 9 written in AspectJ are refactored from the files in version 1 written in Java by means of aspect mining. Fig. 6 and Fig. 7 show the values of RFA and RFP of the same file in version 1 and version 9.

```
- <type kind="class" name="jdbcConnection">
  <metric name="LOCC" value="663" />
  <metric name="RFA" value="67" />
  <metric name="RFM" value="66" />
  <metric name="RFP" value="1" />
</type>
```

**Fig. 6.** RFA In Version 1

```
- <type kind="class" name="jdbcConnection">
  <metric name="LOCC" value="519" />
  <metric name="RFA" value="67" />
  <metric name="RFM" value="63" />
  <metric name="RFP" value="4" />
</type>
```

**Fig. 7.** RFA In Version 9

At the same time, we find that metrics such as NOC, CFA and CMC are increasingly or decreasingly changeable. So these coupling metrics are not always important to the AO software. Among these coupling metrics, because of refactoring from Java to AspectJ and the modification of the access to the private fields, CFA maybe decrease while DIT increase.

Moreover, we use the empirical sample to verify what external attribute is more important to the coupling metrics. After checking the result, we find that the file "ProfilingAbstractAspect.aj" are different according to the date between the different versions. However, after checking the values of coupling metrics of the file "ProfilingAbstractAspect.aj", the result is surprising to us that all the values are the same. So, the external attribute AGE seems to have less impact on the values of the coupling metrics of the files between the different versions. However, it can be seen from the Fig. 6 and Fig. 7 that the values of coupling metrics are changeable because of the different values of LOCC.

Now we can make a conclusion that LOCC is more important to the coupling metrics than AGE. From a deeper viewpoint, we further study what kind of file LOCC affects more. In fact, LOCC has more impact on the files written in Java while LOCC has less on the files written in AspectJ. Since the files written in AspectJ have implemented the crosscut concerns of the Java files, the AspectJ files are stable to the project and less changeable.

## 5 Related Work

Although AOP is a brand new programming methodology and provides a powerful programming method, AOP is currently still at the research stage of development just like when OOP was born. However, many researches in the coupling framework for AO software have provided a strong theoretical support and useful experience.

Ceccato and Tonella [8] have extended OO coupling framework to AO coupling framework in terms of the new AO features and proposed some coupling metrics such as WOM, DIT, and NOC etc. Since this coupling framework is based on the Chidamber and Kemerer's [9] framework, many coupling metrics are still be used in the coupling framework for AO software.

Zhao [16] has proposed a coupling framework for AO system by analyzing the dependencies between the modules of aspects and classes. An aspect may also interact with one or more classed in many ways through advice, inter-type declaration, pointcut and method call. However, the dependencies among aspects and aspects and the dependencies among classes and classes are not be considered.

Further research on AO coupling framework has been carried out by Bartsch and Harrison [4]. This coupling framework differs in the common focus used as the criterion of coupling framework.

Although there are some existing AO coupling frameworks, these frameworks are lack of the support of the metric tool, so the values of metrics proposed in the coupling framework can not be calculated.

Empirical validation of coupling metrics has so far mostly been carried out for OO software. Briand *et al.* [6], for example, validated OO coupling metrics as indicators of error proneness for objects. An initial study into an empirical validation of coupling metrics for AO software has been carried out by Bartsch and Harrison [5]. However, there are only 8 metrics studied in the paper and the metrics about the method invocation are not considered.

## 6 Concluding Remarks

In this paper we presented our evaluation for a coupling metrics suite for AO software. To do so, we first selected a set of coupling metrics, which consists of 16 coupling metrics, for AO software. We then developed a coupling metric tool to calculate these metrics for evaluation. Finally we evaluated these metrics by means of correlation analysis between coupling metrics and external attributes. In our future work, we would like to do more experiment to evaluate these coupling metrics for AspectJ. We also will take the coupling metrics for changes between versions into consideration.

## References

1. Ajhsqldb: <http://sourceforge.net/projects/ajhsqldb/>.

2. AopMetrics: <http://aopmetrics.tigris.org/>.
3. AspectJ: <http://www.eclipse.org/aspectj/>.
4. Bartsch, M., Harrison, R. A Coupling Framework for AspectJ - Extended Abstract. Evaluation and Assessment in Software Engineering (EASE), Keele, UK (2006).
5. Bartsch, M., Harrison, R. "Towards an Empirical Validation of Aspect-Oriented Coupling Metrics," presented at ASAT Workshop at the Aspect-Oriented Software Development Conference (AOSD), Vancouver, BC, 2007.
6. Briand L, Devanbu P, Melo W. An investigation into coupling measures for C++. In: Proc 19th International Conference Software Engineering, ICSE'97, Boston, 1997. 412- 421.
7. Briand, L.C., Daly, J.W., Wust, J.K. A Unified Framework for Coupling Measurement in Object-Oriented Systems. IEEE Transactions on Software Engineering 25 (1999) 91-121.
8. Ceccato, M. and Tonella, P. 'Measuring the Effects of Software Aspectization', CD-Rom Proceedings of the 1st Workshop on Aspect Reverse Engineering (WARE 2004). Nov. 2004. Delft, The Netherlands.
9. Chidamber, R. and Kemerer, C.F. A Metrics Suite for Object-Oriented Design, IEEE Transactions on Software Engineering, 20(6):476-493, 1994.
10. Glassbox: <http://www.glassbox.com/>.
11. Gregor Kiczales, John Lamping, Anurag Menhdhekar, Chris Maeda, Cristina Lopes, Jean-Marc Loingtier and John Irwin, "Aspect-Oriented Programming", Proc. 11th European Conference on Object-Oriented Programming, pp.220-242, 1997.
12. Kitchenham, B.A., Fenton, N. and Pfleeger, S.L., 'Towards a Framework for Software Measurement Validation', IEEE Transactions on Software Engineering, 1995, Vol.21, No.12, pp. 929-944.
13. MuLATO: <http://mulato.sourceforge.net/>.
14. SPSS: <http://www.spss.com/>.
15. WinMerge: <http://winmerge.org/>.
16. Zhao, J. 'Measuring Coupling in Aspect-Oriented Systems', 10th International Software Metrics Symposium (METRICS'2004), (Late Breaking Paper), Chicago, USA, Sept. 14-16, 2004.